

# BNI Hub Deals - Mobile API Documentation

**Version:** 1.3.0

**Base URL:** <https://your-domain.com/api/v1/mobile>

**Last Updated:** November 2024

---

## Table of Contents

1. Quick Reference
  2. Overview
  3. Authentication
  4. Error Handling
  5. Rate Limiting
  6. API Endpoints
    - Health Check
    - Authentication Endpoints
    - Profile Endpoints
    - Deals Endpoints
    - Deal Redemption
    - Wishlist Endpoints
    - Categories Endpoints
    - Push Notifications
  7. Flutter Data Models
  8. API Service Class
  9. Recommended Flutter Packages
  10. Security Best Practices
- 

## Quick Reference

### All API Endpoints at a Glance

Category	Method	Endpoint	Auth	Description
<b>Health</b>	GET	/health	No	Check API status
<b>Auth</b>	POST	/auth/request-otp	No	Request OTP via WhatsApp
<b>Auth</b>	POST	/auth/verify-otp	No	Verify OTP & get JWT token
<b>Auth</b>	GET	/auth/me	Yes	Get current user profile
<b>Profile</b>	PATCH	/profile	Yes	Update user profile

Category	Method	Endpoint	Auth	Description
<b>Deals</b>	GET	/deals	Yes	List all deals (paginated, searchable)
<b>Deals</b>	GET	/deals/premium	Yes	List premium deals only
<b>Deals</b>	GET	/deals/pro-discount	Yes	List PRO discount deals
<b>Deals</b>	GET	/deals/:id	Yes	Get deal details
<b>Redemption</b>	POST	/deals/:id/request-redemption	Yes	Request deal redemption (sends OTP to seller)
<b>Redemption</b>	POST	/deals/:id/verify-redemption	Yes	Verify OTP & complete redemption
<b>Redemption</b>	GET	/redemptions	Yes	Get user's redemption history
<b>Wishlist</b>	GET	/wishlist	Yes	Get user's wishlist
<b>Wishlist</b>	POST	/wishlist	Yes	Add deal to wishlist
<b>Wishlist</b>	DELETE	/wishlist/:dealId	Yes	Remove deal from wishlist
<b>Wishlist</b>	GET	/wishlist/check/:dealId	Yes	Check if deal is in wishlist
<b>Categories</b>	GET	/categories	Yes	List all categories
<b>Push</b>	POST	/fcm/register	Yes	Register FCM token
<b>Push</b>	DELETE	/fcm/unregister	Yes	Unregister FCM token

#### Authentication

- **Token Type:** Bearer JWT
- **Token Validity:** 30 days
- **Header Format:** Authorization: Bearer <token>

#### Rate Limits

- OTP requests: 3/minute per IP
- General API: 100 requests/15 minutes per IP

## Overview

This API provides mobile access to the BNI Hub Deals platform. It allows BNI members to: - Authenticate via WhatsApp OTP - Browse and search deals - View deal details with role-based discounts - Receive push notifications for new deals

## Content Type

All requests must include:

Content-Type: application/json

## Authentication Header

Protected endpoints require a Bearer token:

Authorization: Bearer <your\_jwt\_token>

---

## Authentication

The mobile API uses **WhatsApp OTP authentication**. Members must have their phone number registered in the BNI member database.

## Authentication Flow

1. User enters phone number
2. App calls POST /auth/request-otp
3. User receives OTP via WhatsApp
4. User enters OTP in app
5. App calls POST /auth/verify-otp
6. App receives JWT token (valid for 30 days)
7. App stores token and uses it for all subsequent requests

## Token Expiration

- JWT tokens are valid for **30 days**
  - When a token expires, the user must re-authenticate
  - Store the token securely using Flutter's `flutter_secure_storage` package
- 

## Error Handling

### Error Response Format

All errors follow this structure:

```
{
  "success": false,
  "error": "Human-readable error message"
}
```

## HTTP Status Codes

Code	Description
200	Success
400	Bad Request - Invalid parameters or validation error
401	Unauthorized - Missing or invalid token
403	Forbidden - Token expired or access denied
404	Not Found - Resource does not exist
429	Too Many Requests - Rate limit exceeded
500	Internal Server Error

## Common Error Messages

Error	Cause	Solution
“Phone number not registered”	Phone not in member database	Contact chapter admin
“Your membership is inactive”	Member status is not active	Contact chapter admin
“OTP expired”	OTP older than 10 minutes	Request new OTP
“Too many incorrect attempts”	3+ wrong OTP entries	Request new OTP
“Authentication required”	Missing Bearer token	Include token in header
“Invalid or expired token”	Token is invalid/expired	Re-authenticate

## Rate Limiting

### OTP Requests

- **Limit:** 3 requests per minute per IP
- **Error:** {"success": false, "error": "Too many OTP requests. Please wait 1 minute."}

## General API Requests

- **Limit:** 100 requests per 15 minutes per IP
  - **Error:** {"success": false, "error": "Too many requests. Please try again later."}
- 

## API Endpoints

---

### Health Check

Check if the API is running.

**Endpoint:** GET /health

**Authentication:** Not required

### Response

```
{
  "success": true,
  "version": "1.0.0",
  "timestamp": "2024-11-25T10:30:00.000Z",
  "devMode": false
}
```

### Flutter Example

```
Future<bool> checkApiHealth() async {
  final response = await http.get(
    Uri.parse('$baseUrl/health'),
  );

  if (response.statusCode == 200) {
    final data = jsonDecode(response.body);
    return data['success'] == true;
  }
  return false;
}
```

---

## Authentication Endpoints

---

**Request OTP** Send an OTP to the user's WhatsApp number.

**Endpoint:** POST /auth/request-otp

**Authentication:** Not required

#### Request Body

Field	Type	Required	Description
phone	string	Yes	Phone number (any format accepted)

#### Request Example

```
{  
  "phone": "+971501234567"  
}
```

#### Success Response (200)

```
{  
  "success": true,  
  "message": "OTP sent to your WhatsApp number",  
  "expiresIn": "10 minutes",  
  "memberName": "John Smith"  
}
```

#### Error Responses Phone not registered (404):

```
{  
  "success": false,  
  "error": "Phone number not registered. Please contact your chapter admin."  
}
```

#### Inactive membership (403):

```
{  
  "success": false,  
  "error": "Your membership is inactive. Please contact your chapter admin."  
}
```

#### Rate limited (429):

```
{  
  "success": false,  
  "error": "Too many OTP requests. Please wait 1 minute."  
}
```

## Flutter Example

```
Future<OtpResponse> requestOtp(String phone) async {
  final response = await http.post(
    Uri.parse('$baseUrl/auth/request-otp'),
    headers: {'Content-Type': 'application/json'},
    body: jsonEncode({'phone': phone}),
  );

  final data = jsonDecode(response.body);

  if (response.statusCode == 200 && data['success'] == true) {
    return OtpResponse(
      success: true,
      message: data['message'],
      memberName: data['memberName'],
    );
  } else {
    throw ApiException(data['error'] ?? 'Failed to send OTP');
  }
}
```

---

**Verify OTP** Verify the OTP and receive a JWT token.

**Endpoint:** POST /auth/verify-otp

**Authentication:** Not required

## Request Body

Field	Type	Required	Description
phone	string	Yes	Phone number (same as request-otp)
otp	string	Yes	6-digit OTP received via WhatsApp

## Request Example

```
{
  "phone": "+971501234567",
  "otp": "123456"
}
```

## Success Response (200)

```
{
  "success": true,
```

```

"message": "Login successful",
"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
"user": {
  "id": "uuid-here",
  "name": "John Smith",
  "firstName": "John",
  "lastName": "Smith",
  "phone": "+971501234567",
  "email": "john@example.com",
  "businessName": "Smith Consulting",
  "chapterName": "Dubai Business Chapter",
  "memberRole": "gold",
  "memberType": "pro",
  "isRegionalProfile": false,
  "isGoldMember": true,
  "isCurrentLT": false,
  "isGreenMember": false,
  "profileImage": "https://storage.example.com/profiles/john.jpg",
  "status": "active"
}
}

```

### User Fields

Field	Type	Description
id	string	Unique member identifier (UUID)
name	string	Full name (firstName + lastName)
firstName	string	Member's first name
lastName	string	Member's last name
phone	string	Phone number
email	string	Email address
businessName	string	Business name
chapterName	string	BNI chapter name
memberRole	string	Role: "regular", "leadership", "ro", "green", "gold"
memberType	string	Type: "regular" or "pro"
isRegionalProfile	boolean	PRO option: Regional Profile member
isGoldMember	boolean	PRO option: Gold member
isCurrentLT	boolean	PRO option: Current Leadership Team member
isGreenMember	boolean	PRO option: Green member
profileImage	string	Profile image URL (nullable)
status	string	Membership status



#### Error Responses Invalid OTP (400):

```
{
  "success": false,
  "error": "Invalid OTP",
  "attemptsLeft": 2
}
```

#### OTP expired (400):

```
{
  "success": false,
  "error": "OTP expired. Please request a new one."
}
```

#### Too many attempts (400):

```
{
  "success": false,
  "error": "Too many incorrect attempts. Please request a new OTP."
}
```

#### Flutter Example

```
Future<AuthResponse> verifyOtp(String phone, String otp) async {
  final response = await http.post(
    Uri.parse('$baseUrl/auth/verify-otp'),
    headers: {'Content-Type': 'application/json'},
    body: jsonEncode({'phone': phone, 'otp': otp}),
  );

  final data = jsonDecode(response.body);

  if (response.statusCode == 200 && data['success'] == true) {
    // Store token securely
    await secureStorage.write(key: 'auth_token', value: data['token']);

    return AuthResponse(
      success: true,
      token: data['token'],
      user: User.fromJson(data['user']),
    );
  } else {
    throw ApiException(
      data['error'] ?? 'Verification failed',
      attemptsLeft: data['attemptsLeft'],
    );
  }
}
```

```
}
```

---

**Get Current User Profile** Get the authenticated user's profile.

**Endpoint:** GET /auth/me

**Authentication:** Required

**Success Response (200)**

```
{
  "success": true,
  "user": {
    "id": "uuid-here",
    "name": "John Smith",
    "firstName": "John",
    "lastName": "Smith",
    "phone": "+971501234567",
    "email": "john@example.com",
    "businessName": "Smith Consulting",
    "chapterName": "Dubai Business Chapter",
    "memberRole": "gold",
    "memberType": "pro",
    "isRegionalProfile": false,
    "isGoldMember": true,
    "isCurrentLT": false,
    "isGreenMember": false,
    "profileImage": "https://storage.example.com/profiles/john.jpg",
    "status": "active",
    "membershipEndDate": "2025-12-31"
  }
}
```

See User Fields table above for field descriptions.

### Flutter Example

```
Future<User> getCurrentUser() async {
  final token = await secureStorage.read(key: 'auth_token');

  final response = await http.get(
    Uri.parse('$baseUrl/auth/me'),
    headers: {
      'Content-Type': 'application/json',
      'Authorization': 'Bearer $token',
    },
  );
}
```

```

final data = jsonDecode(response.body);

if (response.statusCode == 200 && data['success'] == true) {
  return User.fromJson(data['user']);
} else {
  throw ApiException(data['error'] ?? 'Failed to get profile');
}
}

```

---

## Profile Endpoints

---

**Edit Profile** Update the authenticated user's profile information.

**Endpoint:** PATCH /profile

**Authentication:** Required

### Request Body

Field	Type	Required	Description
firstName	string	No	First name (cannot be empty if provided)
lastName	string	No	Last name (cannot be empty if provided)
email	string	No	Email address (validated format, cannot be empty if provided)
businessName	string	No	Business name (cannot be empty if provided)
profileImage	string	No	Profile image URL (can be null to clear)

**Note:** Only include fields you want to update. At least one field must be provided. All required fields (firstName, lastName, email, businessName) cannot be cleared once set.

### Request Example

```

{
  "firstName": "John",
  "lastName": "Smith",

```

```

    "email": "john.updated@example.com",
    "businessName": "Smith Consulting Ltd"
  }

```

#### Success Response (200)

```

{
  "success": true,
  "message": "Profile updated successfully",
  "user": {
    "id": "uuid-here",
    "name": "John Smith",
    "firstName": "John",
    "lastName": "Smith",
    "phone": "+971501234567",
    "email": "john.updated@example.com",
    "businessName": "Smith Consulting Ltd",
    "chapterName": "Dubai Business Chapter",
    "memberRole": "gold",
    "memberType": "pro",
    "isRegionalProfile": false,
    "isGoldMember": true,
    "isCurrentLT": false,
    "isGreenMember": false,
    "profileImage": "https://storage.example.com/profiles/john.jpg",
    "status": "active"
  }
}

```

#### Error Responses No fields provided (400):

```

{
  "success": false,
  "error": "No valid fields to update. Allowed fields: firstName, lastName, email, businessName"
}

```

#### Empty first name (400):

```

{
  "success": false,
  "error": "First name cannot be empty"
}

```

#### Invalid email format (400):

```

{
  "success": false,

```

```
    "error": "Invalid email format"
  }
}
```

Empty email (400):

```
{
  "success": false,
  "error": "Email cannot be empty"
}
```

Empty business name (400):

```
{
  "success": false,
  "error": "Business name cannot be empty"
}
```

### Flutter Example

```
Future<User> updateProfile({
  String? firstName,
  String? lastName,
  String? email,
  String? businessName,
  String? profileImage,
}) async {
  final token = await secureStorage.read(key: 'auth_token');

  final Map<String, dynamic> body = {};
  if (firstName != null) body['firstName'] = firstName;
  if (lastName != null) body['lastName'] = lastName;
  if (email != null) body['email'] = email;
  if (businessName != null) body['businessName'] = businessName;
  if (profileImage != null) body['profileImage'] = profileImage;

  final response = await http.patch(
    Uri.parse('$baseUrl/profile'),
    headers: {
      'Content-Type': 'application/json',
      'Authorization': 'Bearer $token',
    },
    body: jsonEncode(body),
  );

  final data = jsonDecode(response.body);

  if (response.statusCode == 200 && data['success'] == true) {
    return User.fromJson(data['user']);
  }
}
```

```

    } else {
        throw ApiException(data['error'] ?? 'Failed to update profile');
    }
}

```

---

## Deals Endpoints

---

**List All Deals** Get a paginated list of active deals.

**Endpoint:** GET /deals

**Authentication:** Required

### Query Parameters

Parameter	Type	Default	Description
page	integer	1	Page number
limit	integer	20	Items per page (max 50)
category	string	-	Filter by category name
search	string	-	Search in deal name, description, member name, keywords

### Request Examples

GET /deals

GET /deals?page=2&limit=10

GET /deals?category=Services

GET /deals?search=consulting

GET /deals?category=Food&search=discount&page=1&limit=15

### Success Response (200)

```

{
  "success": true,
  "data": [
    {
      "id": "uuid-here",
      "dealName": "Business Consultation Special",
      "shortDescription": "20% off business consulting services",
      "longDescription": "Get expert business consultation at a discounted rate for BNI mem

```

```

        "memberName": "John Smith",
        "startDate": "2024-01-01",
        "endDate": "2024-12-31",
        "discountType": "percentage",
        "discountValue": "20.00",
        "hasProDiscount": true,
        "proDiscountType": "percentage",
        "proDiscountValue": "25.00",
        "category": "Services",
        "logoImageUrl": "https://storage.example.com/logos/deal1.jpg",
        "bannerImageUrl": "https://storage.example.com/banners/deal1.jpg",
        "isPremiumDeal": false
    }
],
"pagination": {
    "page": 1,
    "limit": 20,
    "totalItems": 45,
    "totalPages": 3,
    "hasNextPage": true,
    "hasPreviousPage": false
}
}

```

### Field Descriptions

Field	Type	Description
id	string	Unique deal identifier (UUID)
dealName	string	Title of the deal
shortDescription	string	Brief description for list view
longDescription	string	Full description for detail view
memberName	string	Name of the BNI member offering the deal
startDate	string	Deal start date (YYYY-MM-DD)
endDate	string	Deal end date (YYYY-MM-DD)
discountType	string	“percentage” or “flat”
discountValue	string	Discount amount (e.g., “20.00” for 20%)
hasProDiscount	boolean	Whether PRO members get additional discount
proDiscountType	string	PRO member discount type: “percentage” or “flat”
proDiscountValue	string	PRO member discount amount
category	string	Deal category name
logoImageUrl	string	URL of the business logo (nullable)

Field	Type	Description
bannerImageUrl	string	URL of the deal banner image (nullable)
isPremiumDeal	boolean	Whether this is a featured/premium deal

### Flutter Example

```
Future<DealsResponse> getDeals({
  int page = 1,
  int limit = 20,
  String? category,
  String? search,
}) async {
  final token = await secureStorage.read(key: 'auth_token');

  final queryParams = {
    'page': page.toString(),
    'limit': limit.toString(),
    if (category != null) 'category': category,
    if (search != null) 'search': search,
  };

  final uri = Uri.parse('$baseUrl/deals').replace(queryParameters: queryParams);

  final response = await http.get(
    uri,
    headers: {
      'Content-Type': 'application/json',
      'Authorization': 'Bearer $token',
    },
  );

  final data = jsonDecode(response.body);

  if (response.statusCode == 200 && data['success'] == true) {
    return DealsResponse(
      deals: (data['data'] as List).map((d) => Deal.fromJson(d)).toList(),
      pagination: Pagination.fromJson(data['pagination']),
    );
  } else {
    throw ApiException(data['error'] ?? 'Failed to fetch deals');
  }
}
```



---

**Get Premium Deals** Get a paginated list of premium/featured deals only.

**Endpoint:** GET /deals/premium

**Authentication:** Required

#### Query Parameters

Parameter	Type	Default	Description
page	integer	1	Page number
limit	integer	20	Items per page (max 50)

#### Success Response (200)

```
{
  "success": true,
  "data": [
    {
      "id": "uuid-here",
      "dealName": "Premium Business Package",
      "shortDescription": "Exclusive 30% discount on premium services",
      "longDescription": "Our flagship business package...",
      "memberName": "Jane Doe",
      "startDate": "2024-01-01",
      "endDate": "2024-12-31",
      "discountType": "percentage",
      "discountValue": "30.00",
      "hasProDiscount": true,
      "proDiscountType": "percentage",
      "proDiscountValue": "35.00",
      "category": "Services",
      "logoImageUrl": "https://storage.example.com/logos/premium1.jpg",
      "bannerImageUrl": "https://storage.example.com/banners/premium1.jpg",
      "isPremiumDeal": true
    }
  ],
  "pagination": {
    "page": 1,
    "limit": 20,
    "totalItems": 5,
    "totalPages": 1,
    "hasNextPage": false,
    "hasPreviousPage": false
  }
}
```

```

    }
  }
}

```

### Flutter Example

```

Future<DealsResponse> getPremiumDeals({int page = 1, int limit = 20}) async {
  final token = await secureStorage.read(key: 'auth_token');

  final response = await http.get(
    Uri.parse('$baseUrl/deals/premium?page=$page&limit=$limit'),
    headers: {
      'Content-Type': 'application/json',
      'Authorization': 'Bearer $token',
    },
  );

  final data = jsonDecode(response.body);

  if (response.statusCode == 200 && data['success'] == true) {
    return DealsResponse(
      deals: (data['data'] as List).map((d) => Deal.fromJson(d)).toList(),
      pagination: Pagination.fromJson(data['pagination']),
    );
  } else {
    throw ApiException(data['error'] ?? 'Failed to fetch premium deals');
  }
}

```

---

### Get PRO Discount Deals

Get all active deals that have PRO member discount available. These are deals where `hasProDiscount` is `true`, meaning PRO members can get enhanced discount rates.

**Endpoint:** GET `/deals/pro-discount`

**Authentication:** Required

### Query Parameters

Parameter	Type	Default	Description
page	integer	1	Page number
limit	integer	20	Items per page (max 50)
category	string	-	Optional category filter

### Success Response (200)

```
{
  "success": true,
  "data": [
    {
      "id": "uuid-here",
      "dealName": "Executive Coaching Package",
      "shortDescription": "Special discount for PRO members",
      "longDescription": "Comprehensive executive coaching program...",
      "memberName": "John Smith",
      "startDate": "2024-01-01",
      "endDate": "2024-12-31",
      "discountType": "percentage",
      "discountValue": "20.00",
      "hasProDiscount": true,
      "proDiscountType": "percentage",
      "proDiscountValue": "30.00",
      "category": "Services",
      "logoImageUrl": "https://storage.example.com/logos/coaching.jpg",
      "bannerImageUrl": "https://storage.example.com/banners/coaching.jpg",
      "isPremiumDeal": false
    }
  ],
  "pagination": {
    "page": 1,
    "limit": 20,
    "totalItems": 12,
    "totalPages": 1,
    "hasNextPage": false,
    "hasPreviousPage": false
  }
}
```

### Response Fields

Field	Type	Description
discountType	string	Regular discount type (“percentage” or “fixed”)
discountValue	string	Regular discount value
hasProDiscount	boolean	Always <b>true</b> for this endpoint
proDiscountType	string	PRO member discount type
proDiscountValue	string	PRO member discount value

### Flutter Example

```

Future<DealsResponse> getProDiscountDeals({
    int page = 1,
    int limit = 20,
    String? category,
}) async {
    final token = await secureStorage.read(key: 'auth_token');

    String url = '$baseUrl/deals/pro-discount?page=$page&limit=$limit';
    if (category != null) {
        url += '&category=${Uri.encodeComponent(category)}';
    }

    final response = await http.get(
        Uri.parse(url),
        headers: {
            'Content-Type': 'application/json',
            'Authorization': 'Bearer $token',
        },
    );

    final data = jsonDecode(response.body);

    if (response.statusCode == 200 && data['success'] == true) {
        return DealsResponse(
            deals: (data['data'] as List).map((d) => Deal.fromJson(d)).toList(),
            pagination: Pagination.fromJson(data['pagination']),
        );
    } else {
        throw ApiException(data['error'] ?? 'Failed to fetch PRO discount deals');
    }
}

```

## Usage Notes

- This endpoint returns all deals where PRO members can get enhanced discounts
- Regular members will still see these deals with regular discount values
- PRO members (with memberType='pro' and at least one PRO option enabled) will receive the `proDiscountValue` when viewing deal details
- Use this endpoint to display a dedicated “PRO Deals” section in the app

---

**Get Deal Details** Get detailed information about a specific deal.

**Endpoint:** GET /deals/:id

**Authentication:** Required

## Path Parameters

Parameter	Type	Description
id	string	Deal UUID

## Success Response (200)

```
{
  "success": true,
  "data": {
    "id": "uuid-here",
    "dealName": "Business Consultation Special",
    "shortDescription": "20% off business consulting services",
    "longDescription": "Get expert business consultation at a discounted rate for BNI member",
    "memberName": "John Smith",
    "startDate": "2024-01-01",
    "endDate": "2024-12-31",
    "discountType": "percentage",
    "discountValue": "25.00",
    "hasProDiscount": true,
    "category": "Services",
    "logoImageUrl": "https://storage.example.com/logos/deal1.jpg",
    "bannerImageUrl": "https://storage.example.com/banners/deal1.jpg",
    "isPremiumDeal": false
  }
}
```

**PRO Member Discount Logic** The API automatically applies PRO discounts based on the user's member type and PRO options:

**Eligibility Criteria:** A member qualifies for PRO discount if: 1. `memberType` is "pro", AND 2. At least one of the following is true: - `isRegionalProfile` = true - `isGoldMember` = true - `isCurrentLT` = true - `isGreenMember` = true

Member Type	PRO Options	Gets PRO Discount
pro	Any PRO option enabled	Yes
pro	No PRO options enabled	No
regular	N/A	No

When `hasProDiscount` is `true`, the `discountType` and `discountValue` fields already reflect the enhanced PRO discount for the authenticated user.

### Error Response (404)

```
{
  "success": false,
  "error": "Deal not found"
}
```

### Flutter Example

```
Future<Deal> getDealDetails(String dealId) async {
  final token = await secureStorage.read(key: 'auth_token');

  final response = await http.get(
    Uri.parse('$baseUrl/deals/$dealId'),
    headers: {
      'Content-Type': 'application/json',
      'Authorization': 'Bearer $token',
    },
  );

  final data = jsonDecode(response.body);

  if (response.statusCode == 200 && data['success'] == true) {
    return Deal.fromJson(data['data']);
  } else if (response.statusCode == 404) {
    throw NotFoundException('Deal not found');
  } else {
    throw ApiException(data['error'] ?? 'Failed to fetch deal details');
  }
}
```

---

### Deal Redemption

The deal redemption system allows members to “get” deals through an OTP verification process. When a user wants to redeem a deal, the deal owner receives an OTP via WhatsApp. The user must enter this OTP (shared offline by the deal owner) to complete the redemption.

### Redemption Flow

1. User views a deal and clicks "Get the Deal"
2. App calls POST /deals/:id/request-redemption
3. Deal owner receives OTP via WhatsApp
4. Deal owner shares OTP with user (offline/in-person)
5. User enters OTP in the app
6. App calls POST /deals/:id/verify-redemption

## 7. If valid, redemption is recorded with savings details

---

**Request Deal Redemption** Initiate a deal redemption by sending OTP to the deal owner.

**Endpoint:** POST /deals/:id/request-redemption

**Authentication:** Required

**Rate Limit:** 3 requests per minute

### Path Parameters

Parameter	Type	Description
id	string	Deal UUID

### Success Response (200)

```
{
  "success": true,
  "message": "OTP sent to deal owner. Please ask them for the code.",
  "data": {
    "redemptionId": "uuid-here",
    "dealName": "Business Consultation Special",
    "sellerName": "John Smith",
    "expiresAt": "2024-01-15T10:30:00.000Z",
    "expiresInMinutes": 15,
    "devBypass": false
  }
}
```

### Error Responses

Status	Error	Description
404	Deal not found or inactive	Deal doesn't exist or is not active
400	Deal owner not found	Deal has no associated member
400	Deal owner is inactive	Deal owner's membership is inactive
400	You cannot redeem your own deal	Buyer cannot redeem their own deals

Status	Error	Description
500	Failed to send OTP to deal owner	WhatsApp message failed

### Flutter Example

```
Future<RedemptionRequest> requestRedemption(String dealId) async {
  final token = await secureStorage.read(key: 'auth_token');

  final response = await http.post(
    Uri.parse('$baseUrl/deals/$dealId/request-redemption'),
    headers: {
      'Content-Type': 'application/json',
      'Authorization': 'Bearer $token',
    },
  );

  final data = jsonDecode(response.body);

  if (response.statusCode == 200 && data['success'] == true) {
    return RedemptionRequest.fromJson(data['data']);
  } else {
    throw ApiException(data['error'] ?? 'Failed to request redemption');
  }
}
```

---

**Verify Redemption OTP** Complete the deal redemption by verifying the OTP shared by the deal owner.

**Endpoint:** POST /deals/:id/verify-redemption

**Authentication:** Required

### Path Parameters

Parameter	Type	Description
id	string	Deal UUID

### Request Body



Field	Type	Required	Description
otp	string	Yes	6-digit OTP received from deal owner
originalPrice	string	No	Original price before discount
finalPrice	string	No	Final price after discount

```
{
  "otp": "123456",
  "originalPrice": "100.00",
  "finalPrice": "80.00"
}
```

### Success Response (200)

```
{
  "success": true,
  "message": "Deal redeemed successfully!",
  "data": {
    "redemptionId": "uuid-here",
    "dealName": "Business Consultation Special",
    "sellerName": "John Smith",
    "discountType": "percentage",
    "discountValue": "20.00",
    "amountSaved": "20.00",
    "originalPrice": "100.00",
    "finalPrice": "80.00",
    "redeemedAt": "2024-01-15T10:25:00.000Z"
  }
}
```

### Error Responses

Status	Error	Description
400	OTP is required	Missing OTP in request body
400	No pending redemption found	No active OTP request for this deal
400	OTP has expired	OTP validity period (15 min) exceeded
400	Too many failed attempts	Max 3 attempts exceeded
400	Invalid OTP. X attempt(s) remaining	Wrong OTP entered

## Flutter Example

```
Future<Redemption> verifyRedemption({
  required String dealId,
  required String otp,
  double? originalPrice,
  double? finalPrice,
}) async {
  final token = await secureStorage.read(key: 'auth_token');

  final body = {
    'otp': otp,
    if (originalPrice != null) 'originalPrice': originalPrice.toStringAsFixed(2),
    if (finalPrice != null) 'finalPrice': finalPrice.toStringAsFixed(2),
  };

  final response = await http.post(
    Uri.parse('$baseUrl/deals/$dealId/verify-redemption'),
    headers: {
      'Content-Type': 'application/json',
      'Authorization': 'Bearer $token',
    },
    body: jsonEncode(body),
  );

  final data = jsonDecode(response.body);

  if (response.statusCode == 200 && data['success'] == true) {
    return Redemption.fromJson(data['data']);
  } else {
    throw ApiException(data['error'] ?? 'Failed to verify redemption');
  }
}
```

---

**Get Redemption History** Get a list of all deals redeemed by the authenticated user.

**Endpoint:** GET /redemptions

**Authentication:** Required

### Query Parameters

Parameter	Type	Default	Description
page	integer	1	Page number

Parameter	Type	Default	Description
limit	integer	20	Items per page (max 50)

### Success Response (200)

```
{
  "success": true,
  "data": [
    {
      "id": "uuid-here",
      "dealId": "deal-uuid",
      "dealName": "Business Consultation Special",
      "sellerName": "John Smith",
      "discountType": "percentage",
      "discountValue": "20.00",
      "amountSaved": "20.00",
      "originalPrice": "100.00",
      "finalPrice": "80.00",
      "redeemedAt": "2024-01-15T10:25:00.000Z"
    }
  ],
  "pagination": {
    "page": 1,
    "limit": 20,
    "totalItems": 5,
    "totalPages": 1,
    "hasNextPage": false,
    "hasPreviousPage": false
  }
}
```

### Flutter Example

```
Future<RedemptionsResponse> getRedemptionHistory({
  int page = 1,
  int limit = 20,
}) async {
  final token = await secureStorage.read(key: 'auth_token');

  final response = await http.get(
    Uri.parse('$baseUrl/redemptions?page=$page&limit=$limit'),
    headers: {
      'Content-Type': 'application/json',
      'Authorization': 'Bearer $token',
    },
  ),
```

```

);

final data = jsonDecode(response.body);

if (response.statusCode == 200 && data['success'] == true) {
  return RedemptionsResponse(
    redemptions: (data['data'] as List).map((r) => Redemption.fromJson(r)).toList(),
    pagination: Pagination.fromJson(data['pagination']),
  );
} else {
  throw ApiException(data['error'] ?? 'Failed to fetch redemption history');
}
}

```

### Flutter Model Classes

```

class RedemptionRequest {
  final String redemptionId;
  final String dealName;
  final String sellerName;
  final DateTime expiresAt;
  final int expiresInMinutes;
  final bool devBypass;

  RedemptionRequest({
    required this.redemptionId,
    required this.dealName,
    required this.sellerName,
    required this.expiresAt,
    required this.expiresInMinutes,
    required this.devBypass,
  });

  factory RedemptionRequest.fromJson(Map<String, dynamic> json) {
    return RedemptionRequest(
      redemptionId: json['redemptionId'],
      dealName: json['dealName'],
      sellerName: json['sellerName'],
      expiresAt: DateTime.parse(json['expiresAt']),
      expiresInMinutes: json['expiresInMinutes'],
      devBypass: json['devBypass'] ?? false,
    );
  }
}

class Redemption {

```

```

final String id;
final String? dealId;
final String dealName;
final String sellerName;
final String discountType;
final String discountValue;
final String? amountSaved;
final String? originalPrice;
final String? finalPrice;
final DateTime redeemedAt;

Redemption({
  required this.id,
  this.dealId,
  required this.dealName,
  required this.sellerName,
  required this.discountType,
  required this.discountValue,
  this.amountSaved,
  this.originalPrice,
  this.finalPrice,
  required this.redeemedAt,
});

factory Redemption.fromJson(Map<String, dynamic> json) {
  return Redemption(
    id: json['id'] ?? json['redemptionId'],
    dealId: json['dealId'],
    dealName: json['dealName'],
    sellerName: json['sellerName'],
    discountType: json['discountType'],
    discountValue: json['discountValue'],
    amountSaved: json['amountSaved'],
    originalPrice: json['originalPrice'],
    finalPrice: json['finalPrice'],
    redeemedAt: DateTime.parse(json['redeemedAt']),
  );
}
}

```

---

## Wishlist Endpoints

The wishlist allows members to save deals they're interested in for later viewing.

---

**Get Wishlist** Get the authenticated user's wishlist with full deal details.

**Endpoint:** GET /wishlist

**Authentication:** Required

#### Query Parameters

Parameter	Type	Default	Description
page	integer	1	Page number
limit	integer	20	Items per page (max 50)

#### Success Response (200)

```
{
  "success": true,
  "data": [
    {
      "wishlistId": "wishlist-uuid",
      "addedAt": "2024-11-30T10:30:00.000Z",
      "deal": {
        "id": "deal-uuid",
        "dealName": "Business Consultation Special",
        "shortDescription": "Get 25% off on business consulting",
        "longDescription": "Full description here...",
        "memberName": "John Smith",
        "startDate": "2024-01-01",
        "endDate": "2024-12-31",
        "discountType": "percentage",
        "discountValue": "25.00",
        "hasProDiscount": true,
        "proDiscountType": "percentage",
        "proDiscountValue": "35.00",
        "category": "Services",
        "logoImageUrl": "https://storage.example.com/deals/logo.jpg",
        "bannerImageUrl": "https://storage.example.com/deals/banner.jpg",
        "isPremiumDeal": true,
        "status": "active"
      }
    }
  ],
  "pagination": {
    "page": 1,
    "limit": 20,
    "totalItems": 5,
    "totalPages": 1,
  }
}
```

```

        "hasNextPage": false,
        "hasPreviousPage": false
    }
}

```

### Flutter Example

```

Future<WishlistResponse> getWishlist({int page = 1, int limit = 20}) async {
    final token = await secureStorage.read(key: 'auth_token');

    final response = await http.get(
        Uri.parse('$baseUrl/wishlist?page=$page&limit=$limit'),
        headers: {
            'Content-Type': 'application/json',
            'Authorization': 'Bearer $token',
        },
    );

    final data = jsonDecode(response.body);

    if (response.statusCode == 200 && data['success'] == true) {
        return WishlistResponse(
            items: (data['data'] as List).map((item) => WishlistItem.fromJson(item)).toList(),
            pagination: Pagination.fromJson(data['pagination']),
        );
    } else {
        throw ApiException(data['error'] ?? 'Failed to get wishlist');
    }
}

```

---

**Add to Wishlist** Add a deal to the authenticated user's wishlist.

**Endpoint:** POST /wishlist

**Authentication:** Required

### Request Body

Field	Type	Required	Description
dealId	string	Yes	UUID of the deal to add

### Request Example

```
{
  "dealId": "deal-uuid-here"
}
```

#### Success Response (201)

```
{
  "success": true,
  "message": "Deal added to wishlist",
  "data": {
    "wishlistId": "wishlist-uuid",
    "dealId": "deal-uuid",
    "addedAt": "2024-11-30T10:30:00.000Z"
  }
}
```

#### Error Responses Deal already in wishlist (409):

```
{
  "success": false,
  "error": "Deal already in wishlist",
  "wishlistId": "existing-wishlist-uuid"
}
```

#### Deal not found (404):

```
{
  "success": false,
  "error": "Deal not found"
}
```

#### Flutter Example

```
Future<WishlistItem> addToWishlist(String dealId) async {
  final token = await secureStorage.read(key: 'auth_token');

  final response = await http.post(
    Uri.parse('$baseUrl/wishlist'),
    headers: {
      'Content-Type': 'application/json',
      'Authorization': 'Bearer $token',
    },
    body: jsonEncode({'dealId': dealId}),
  );

  final data = jsonDecode(response.body);

  if (response.statusCode == 201 && data['success'] == true) {
```



```

        return WishlistItem(
            wishlistId: data['data']['wishlistId'],
            dealId: data['data']['dealId'],
            addedAt: DateTime.parse(data['data']['addedAt']),
        );
    } else if (response.statusCode == 409) {
        // Deal already in wishlist - return existing wishlist ID
        throw AlreadyInWishlistException(data['wishlistId']);
    } else {
        throw ApiException(data['error'] ?? 'Failed to add to wishlist');
    }
}

```

---

**Remove from Wishlist** Remove a deal from the authenticated user's wishlist.

**Endpoint:** DELETE /wishlist/:dealId

**Authentication:** Required

#### Path Parameters

Parameter	Type	Description
dealId	string	UUID of the deal to remove

#### Success Response (200)

```

{
  "success": true,
  "message": "Deal removed from wishlist"
}

```

#### Error Responses Deal not in wishlist (404):

```

{
  "success": false,
  "error": "Deal not found in wishlist"
}

```

#### Flutter Example

```

Future<void> removeFromWishlist(String dealId) async {
    final token = await secureStorage.read(key: 'auth_token');

    final response = await http.delete(

```

```

Uri.parse('$baseUrl/wishlist/$dealId'),
headers: {
  'Content-Type': 'application/json',
  'Authorization': 'Bearer $token',
},
);

final data = jsonDecode(response.body);

if (response.statusCode != 200 || data['success'] != true) {
  throw ApiException(data['error'] ?? 'Failed to remove from wishlist');
}
}

```

---

**Check Wishlist Status** Check if a specific deal is in the authenticated user's wishlist.

**Endpoint:** GET /wishlist/check/:dealId

**Authentication:** Required

#### Path Parameters

Parameter	Type	Description
dealId	string	UUID of the deal to check

#### Success Response (200)

```

{
  "success": true,
  "inWishlist": true,
  "wishlistId": "wishlist-uuid"
}

```

Or if not in wishlist:

```

{
  "success": true,
  "inWishlist": false,
  "wishlistId": null
}

```

#### Flutter Example

```

Future<bool> isInWishlist(String dealId) async {
  final token = await secureStorage.read(key: 'auth_token');

```

```

final response = await http.get(
  Uri.parse('$baseUrl/wishlist/check/$dealId'),
  headers: {
    'Content-Type': 'application/json',
    'Authorization': 'Bearer $token',
  },
);

final data = jsonDecode(response.body);

if (response.statusCode == 200 && data['success'] == true) {
  return data['inWishlist'] == true;
} else {
  throw ApiException(data['error'] ?? 'Failed to check wishlist status');
}
}

```

---

### Wishlist Flutter Model

```

class WishlistItem {
  final String wishlistId;
  final DateTime addedAt;
  final Deal deal;

  WishlistItem({
    required this.wishlistId,
    required this.addedAt,
    required this.deal,
  });

  factory WishlistItem.fromJson(Map<String, dynamic> json) {
    return WishlistItem(
      wishlistId: json['wishlistId'],
      addedAt: DateTime.parse(json['addedAt']),
      deal: Deal.fromJson(json['deal']),
    );
  }
}

class AlreadyInWishlistException implements Exception {
  final String wishlistId;
  AlreadyInWishlistException(this.wishlistId);
}

```

---

## Categories Endpoints

---

**List All Categories** Get all active categories.

**Endpoint:** GET /categories

**Authentication:** Required

### Success Response (200)

```
{
  "success": true,
  "data": [
    {
      "id": "uuid-here",
      "name": "Automotive",
      "description": "Car sales, repairs, and automotive services",
      "image": "https://storage.example.com/categories/automotive.jpg"
    },
    {
      "id": "uuid-here",
      "name": "Food & Beverage",
      "description": "Restaurants, catering, and food services",
      "image": "https://storage.example.com/categories/food.jpg"
    },
    {
      "id": "uuid-here",
      "name": "Services",
      "description": "Professional and business services",
      "image": null
    }
  ]
}
```

### Field Descriptions

Field	Type	Description
id	string	Unique category identifier (UUID)
name	string	Category name
description	string	Category description (nullable)
image	string	Category image URL (nullable)

## Flutter Example

```
Future<List<Category>> getCategories() async {
  final token = await secureStorage.read(key: 'auth_token');

  final response = await http.get(
    Uri.parse('$baseUrl/categories'),
    headers: {
      'Content-Type': 'application/json',
      'Authorization': 'Bearer $token',
    },
  );

  final data = jsonDecode(response.body);

  if (response.statusCode == 200 && data['success'] == true) {
    return (data['data'] as List)
      .map((c) => Category.fromJson(c))
      .toList();
  } else {
    throw ApiException(data['error'] ?? 'Failed to fetch categories');
  }
}
```

---

## Push Notifications

---

**Register FCM Token** Register a Firebase Cloud Messaging token for push notifications.

**Endpoint:** POST /fcm/register

**Authentication:** Required

### Request Body

Field	Type	Required	Description
token	string	Yes	FCM device token
deviceType	string	No	“android” or “ios”

### Request Example

```
{
  "token": "dGhpcyBpcyBhIHNhbXBsZSBGQQ00gdG9rZW4...",
}
```

```
    "deviceType": "android"
  }
```

#### Success Response (200)

```
{
  "success": true,
  "message": "FCM token registered successfully"
}
```

#### Flutter Example

```
Future<void> registerFcmToken() async {
  final authToken = await secureStorage.read(key: 'auth_token');
  final fcmToken = await FirebaseMessaging.instance.getToken();

  if (fcmToken == null) return;

  final response = await http.post(
    Uri.parse('$baseUrl/fcm/register'),
    headers: {
      'Content-Type': 'application/json',
      'Authorization': 'Bearer $authToken',
    },
    body: jsonEncode({
      'token': fcmToken,
      'deviceType': Platform.isAndroid ? 'android' : 'ios',
    }),
  );

  final data = jsonDecode(response.body);

  if (response.statusCode != 200 || data['success'] != true) {
    print('Failed to register FCM token: ${data['error']}');
  }
}
```

---

**Unregister FCM Token** Remove an FCM token (e.g., on logout).

**Endpoint:** DELETE /fcm/unregister

**Authentication:** Required

#### Request Body

Field	Type	Required	Description
token	string	Yes	FCM device token to remove

### Request Example

```
{
  "token": "dGhpcyBpcyBhIHNhbXBsZSBGQQ00gdG9rZW4..."
}
```

### Success Response (200)

```
{
  "success": true,
  "message": "FCM token unregistered successfully"
}
```

### Flutter Example

```
Future<void> unregisterFcmToken() async {
  final authToken = await secureStorage.read(key: 'auth_token');
  final fcmToken = await FirebaseMessaging.instance.getToken();

  if (fcmToken == null) return;

  final response = await http.delete(
    Uri.parse('$baseUrl/fcm/unregister'),
    headers: {
      'Content-Type': 'application/json',
      'Authorization': 'Bearer $authToken',
    },
    body: jsonEncode({'token': fcmToken}),
  );

  // Clear local auth token
  await secureStorage.delete(key: 'auth_token');
}
```

## Flutter Data Models

Here are example Dart model classes for your Flutter app:

```
// user.dart
class User {
  final String id;
```

```

final String name;
final String firstName;
final String lastName;
final String phone;
final String? email;
final String? businessName;
final String chapterName;
final String memberRole;
final String memberType; // 'regular' or 'pro'
final bool isRegionalProfile;
final bool isGoldMember;
final bool isCurrentLT;
final bool isGreenMember;
final String? profileImage;
final String status;
final String? membershipEndDate;

User({
  required this.id,
  required this.name,
  required this.firstName,
  required this.lastName,
  required this.phone,
  this.email,
  this.businessName,
  required this.chapterName,
  required this.memberRole,
  required this.memberType,
  required this.isRegionalProfile,
  required this.isGoldMember,
  required this.isCurrentLT,
  required this.isGreenMember,
  this.profileImage,
  required this.status,
  this.membershipEndDate,
});

factory User.fromJson(Map<String, dynamic> json) {
  return User(
    id: json['id'],
    name: json['name'],
    firstName: json['firstName'] ?? '',
    lastName: json['lastName'] ?? '',
    phone: json['phone'],
    email: json['email'],
    businessName: json['businessName'],

```



```

        chapterName: json['chapterName'],
        memberRole: json['memberRole'],
        memberType: json['memberType'] ?? 'regular',
        isRegionalProfile: json['isRegionalProfile'] ?? false,
        isGoldMember: json['isGoldMember'] ?? false,
        isCurrentLT: json['isCurrentLT'] ?? false,
        isGreenMember: json['isGreenMember'] ?? false,
        profileImage: json['profileImage'],
        status: json['status'],
        membershipEndDate: json['membershipEndDate'],
    );
}

/// Check if user qualifies for PRO discounts
bool get isPROMember {
    return memberType == 'pro' &&
        (isRegionalProfile || isGoldMember || isCurrentLT || isGreenMember);
}

}

// deal.dart
class Deal {
    final String id;
    final String dealName;
    final String shortDescription;
    final String? longDescription;
    final String memberName;
    final String? startDate;
    final String? endDate;
    final String discountType;
    final String discountValue;
    final bool? hasProDiscount;
    final String? proDiscountType;
    final String? proDiscountValue;
    final String category;
    final String? logoImageUrl;
    final String? bannerImageUrl;
    final bool isPremiumDeal;
    final String? status;

    Deal({
        required this.id,
        required this.dealName,
        required this.shortDescription,
        this.longDescription,
        required this.memberName,
    })

```

```

        this.startDate,
        this.endDate,
        required this.discountType,
        required this.discountValue,
        this.hasProDiscount,
        this.proDiscountType,
        this.proDiscountValue,
        required this.category,
        this.logoImageUrl,
        this.bannerImageUrl,
        required this.isPremiumDeal,
        this.status,
    });

    factory Deal.fromJson(Map<String, dynamic> json) {
        return Deal(
            id: json['id'],
            dealName: json['dealName'],
            shortDescription: json['shortDescription'],
            longDescription: json['longDescription'],
            memberName: json['memberName'],
            startDate: json['startDate'],
            endDate: json['endDate'],
            discountType: json['discountType'],
            discountValue: json['discountValue'],
            hasProDiscount: json['hasProDiscount'],
            proDiscountType: json['proDiscountType'],
            proDiscountValue: json['proDiscountValue'],
            category: json['category'],
            logoImageUrl: json['logoImageUrl'],
            bannerImageUrl: json['bannerImageUrl'],
            isPremiumDeal: json['isPremiumDeal'] ?? false,
            status: json['status'],
        );
    }

    String get formattedDiscount {
        if (discountType == 'percentage') {
            return '${discountValue}% OFF';
        } else {
            return 'AED ${discountValue} OFF';
        }
    }

    String get formattedProDiscount {
        if (hasProDiscount != true || proDiscountType == null || proDiscountValue == null) {

```

```

        return formattedDiscount;
    }
    if (proDiscountType == 'percentage') {
        return '${proDiscountValue}% OFF';
    } else {
        return 'AED ${proDiscountValue} OFF';
    }
}
}

// category.dart
class Category {
    final String id;
    final String name;
    final String? description;
    final String? image;

    Category({
        required this.id,
        required this.name,
        this.description,
        this.image,
    });

    factory Category.fromJson(Map<String, dynamic> json) {
        return Category(
            id: json['id'],
            name: json['name'],
            description: json['description'],
            image: json['image'],
        );
    }
}

// pagination.dart
class Pagination {
    final int page;
    final int limit;
    final int totalItems;
    final int totalPages;
    final bool hasNextPage;
    final bool hasPreviousPage;

    Pagination({
        required this.page,
        required this.limit,

```

```

        required this.totalItems,
        required this.totalPages,
        required this.hasNextPage,
        required this.hasPreviousPage,
    });

    factory Pagination.fromJson(Map<String, dynamic> json) {
        return Pagination(
            page: json['page'],
            limit: json['limit'],
            totalItems: json['totalItems'],
            totalPages: json['totalPages'],
            hasNextPage: json['hasNextPage'],
            hasPreviousPage: json['hasPreviousPage'],
        );
    }
}

```

---

## API Service Class

Here's a complete API service class for Flutter:

```

// api_service.dart
import 'dart:convert';
import 'dart:io';
import 'package:http/http.dart' as http;
import 'package:flutter_secure_storage/flutter_secure_storage.dart';

class ApiService {
    static const String baseUrl = 'https://your-domain.com/api/v1/mobile';
    final FlutterSecureStorage _storage = const FlutterSecureStorage();

    Future<String?> get _authToken => _storage.read(key: 'auth_token');

    Future<Map<String, String>> get _headers async {
        final token = await _authToken;
        return {
            'Content-Type': 'application/json',
            if (token != null) 'Authorization': 'Bearer $token',
        };
    }

    // Health Check
    Future<bool> checkHealth() async {
        try {

```

```

        final response = await http.get(Uri.parse('$baseUrl/health'));
        return response.statusCode == 200;
      } catch (e) {
        return false;
      }
    }

    // Request OTP
    Future<Map<String, dynamic>> requestOtp(String phone) async {
      final response = await http.post(
        Uri.parse('$baseUrl/auth/request-otp'),
        headers: {'Content-Type': 'application/json'},
        body: jsonEncode({'phone': phone}),
      );
      return _handleResponse(response);
    }

    // Verify OTP
    Future<Map<String, dynamic>> verifyOtp(String phone, String otp) async {
      final response = await http.post(
        Uri.parse('$baseUrl/auth/verify-otp'),
        headers: {'Content-Type': 'application/json'},
        body: jsonEncode({'phone': phone, 'otp': otp}),
      );

      final data = _handleResponse(response);
      if (data['success'] == true && data['token'] != null) {
        await _storage.write(key: 'auth_token', value: data['token']);
      }
      return data;
    }

    // Get Current User
    Future<Map<String, dynamic>> getCurrentUser() async {
      final response = await http.get(
        Uri.parse('$baseUrl/auth/me'),
        headers: await _headers,
      );
      return _handleResponse(response);
    }

    // Get Deals
    Future<Map<String, dynamic>> getDeals({
      int page = 1,
      int limit = 20,
      String? category,

```

```

        String? search,
    }) async {
        final queryParams = {
            'page': page.toString(),
            'limit': limit.toString(),
            if (category != null) 'category': category,
            if (search != null) 'search': search,
        };

        final uri = Uri.parse('$baseUrl/deals').replace(queryParameters: queryParams);
        final response = await http.get(uri, headers: await _headers);
        return _handleResponse(response);
    }

    // Get Premium Deals
    Future<Map<String, dynamic>> getPremiumDeals({int page = 1, int limit = 20}) async {
        final response = await http.get(
            Uri.parse('$baseUrl/deals/premium?page=$page&limit=$limit'),
            headers: await _headers,
        );
        return _handleResponse(response);
    }

    // Get Deal Details
    Future<Map<String, dynamic>> getDealDetails(String dealId) async {
        final response = await http.get(
            Uri.parse('$baseUrl/deals/$dealId'),
            headers: await _headers,
        );
        return _handleResponse(response);
    }

    // Get PRO Discount Deals
    Future<Map<String, dynamic>> getProDiscountDeals({
        int page = 1,
        int limit = 20,
        String? category,
    }) async {
        String url = '$baseUrl/deals/pro-discount?page=$page&limit=$limit';
        if (category != null) {
            url += '&category=${Uri.encodeComponent(category)}';
        }
        final response = await http.get(Uri.parse(url), headers: await _headers);
        return _handleResponse(response);
    }
}

```

```

// Update Profile
Future<Map<String, dynamic>> updateProfile({
    String? firstName,
    String? lastName,
    String? email,
    String? businessName,
    String? profileImage,
}) async {
    final Map<String, dynamic> body = {};
    if (firstName != null) body['firstName'] = firstName;
    if (lastName != null) body['lastName'] = lastName;
    if (email != null) body['email'] = email;
    if (businessName != null) body['businessName'] = businessName;
    if (profileImage != null) body['profileImage'] = profileImage;

    final response = await http.patch(
        Uri.parse('$baseUrl/profile'),
        headers: await _headers,
        body: jsonEncode(body),
    );
    return _handleResponse(response);
}

// Request Deal Redemption
Future<Map<String, dynamic>> requestRedemption(String dealId) async {
    final response = await http.post(
        Uri.parse('$baseUrl/deals/$dealId/request-redemption'),
        headers: await _headers,
    );
    return _handleResponse(response);
}

// Verify Deal Redemption
Future<Map<String, dynamic>> verifyRedemption({
    required String dealId,
    required String otp,
    double? originalPrice,
    double? finalPrice,
}) async {
    final body = {
        'otp': otp,
        if (originalPrice != null) 'originalPrice': originalPrice.toStringAsFixed(2),
        if (finalPrice != null) 'finalPrice': finalPrice.toStringAsFixed(2),
    };

    final response = await http.post(

```

```

        Uri.parse('$baseUrl/deals/$dealId/verify-redemption'),
        headers: await _headers,
        body: jsonEncode(body),
    );
    return _handleResponse(response);
}

// Get Redemption History
Future<Map<String, dynamic>> getRedemptionHistory({int page = 1, int limit = 20}) async {
    final response = await http.get(
        Uri.parse('$baseUrl/redemptions?page=$page&limit=$limit'),
        headers: await _headers,
    );
    return _handleResponse(response);
}

// Get Wishlist
Future<Map<String, dynamic>> getWishlist({int page = 1, int limit = 20}) async {
    final response = await http.get(
        Uri.parse('$baseUrl/wishlist?page=$page&limit=$limit'),
        headers: await _headers,
    );
    return _handleResponse(response);
}

// Add to Wishlist
Future<Map<String, dynamic>> addToWishlist(String dealId) async {
    final response = await http.post(
        Uri.parse('$baseUrl/wishlist'),
        headers: await _headers,
        body: jsonEncode({'dealId': dealId}),
    );
    return _handleResponse(response);
}

// Remove from Wishlist
Future<void> removeFromWishlist(String dealId) async {
    final response = await http.delete(
        Uri.parse('$baseUrl/wishlist/$dealId'),
        headers: await _headers,
    );
    _handleResponse(response);
}

// Check Wishlist Status
Future<Map<String, dynamic>> checkWishlistStatus(String dealId) async {

```



```

        final response = await http.get(
            Uri.parse('$baseUrl/wishlist/check/$dealId'),
            headers: await _headers,
        );
        return _handleResponse(response);
    }

    // Get Categories
    Future<Map<String, dynamic>> getCategories() async {
        final response = await http.get(
            Uri.parse('$baseUrl/categories'),
            headers: await _headers,
        );
        return _handleResponse(response);
    }

    // Register FCM Token
    Future<void> registerFcmToken(String fcmToken, String deviceType) async {
        final response = await http.post(
            Uri.parse('$baseUrl/fcm/register'),
            headers: await _headers,
            body: jsonEncode({
                'token': fcmToken,
                'deviceType': deviceType,
            }),
        );
        _handleResponse(response);
    }

    // Unregister FCM Token
    Future<void> unregisterFcmToken(String fcmToken) async {
        final response = await http.delete(
            Uri.parse('$baseUrl/fcm/unregister'),
            headers: await _headers,
            body: jsonEncode({'token': fcmToken}),
        );
        _handleResponse(response);
    }

    // Logout
    Future<void> logout(String? fcmToken) async {
        if (fcmToken != null) {
            await unregisterFcmToken(fcmToken);
        }
        await _storage.delete(key: 'auth_token');
    }

```

```

// Handle Response
Map<String, dynamic> _handleResponse(http.Response response) {
  final data = jsonDecode(response.body);

  if (response.statusCode == 401 || response.statusCode == 403) {
    throw UnauthorizedException(data['error'] ?? 'Authentication required');
  }

  if (response.statusCode == 429) {
    throw RateLimitException(data['error'] ?? 'Too many requests');
  }

  if (response.statusCode >= 400) {
    throw ApiException(data['error'] ?? 'Request failed');
  }

  return data;
}

// Custom Exceptions
class ApiException implements Exception {
  final String message;
  ApiException(this.message);

  @override
  String toString() => message;
}

class UnauthorizedException extends ApiException {
  UnauthorizedException(String message) : super(message);
}

class RateLimitException extends ApiException {
  RateLimitException(String message) : super(message);
}

```

---

## Recommended Flutter Packages

```

# pubspec.yaml
dependencies:
  http: ^1.1.0
  flutter_secure_storage: ^9.0.0

```

```
firebase_messaging: ^14.7.0
firebase_core: ^2.24.0
```

---

## Security Best Practices

1. **Store tokens securely** using `flutter_secure_storage`
  2. **Handle token expiration** gracefully - redirect to login on 401/403
  3. **Use HTTPS** for all API calls
  4. **Validate certificates** in production
  5. **Don't log sensitive data** like tokens or OTPs
- 

## Support

For API issues or questions, contact the BNI Hub Deals development team.